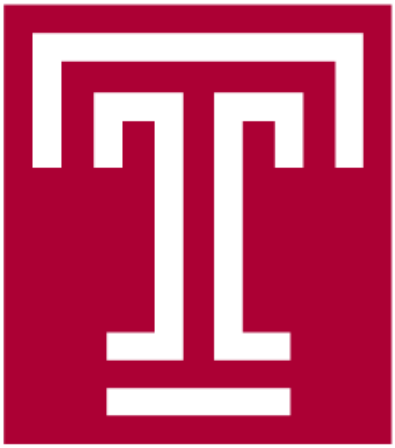
Temple University

College of Engineering

Department of Electrical and Computer Engineering (ECE)

Student Report Cover Page



Course Number: ECE 3412

Lab 6: PID feedback control of a DC motor using experimental set-up | Man Vs. Machine

Student Name: Devin Trejo; Robert Irwin

TUid: 914924557; 914980083

Due Date: 3/24/2015

TA Name: Michael Korostelev

Grade: / 100

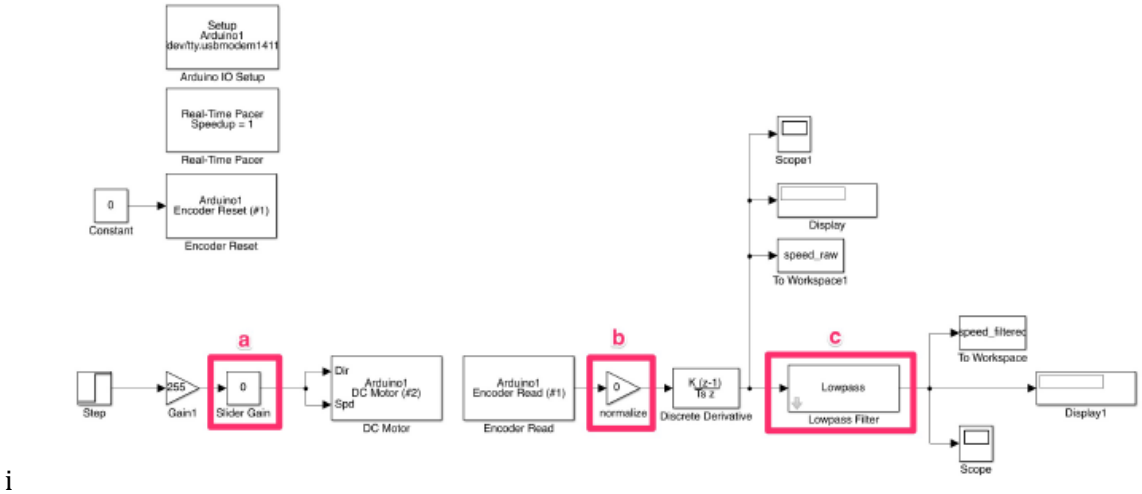
I. Introduction

In Lab 6 we go head to head and challenge ourselves to see if we can manually control the feedback network better than a PI controller can. In our Man vs Machine challenge we are controlling the speed of a DC motor. Our system is built inside MatLab's Simulink toolbox interfacing with an Arduino.

In the previous lab assignment we introduced PID controllers which minimizing the error of the system. P is the proportional gain, I is the integral gain, and D is the derivative gain. A summation of all PID gains is fed back into the motor model. In our manual control side of the lab we will have to adjust a slider adjusting the gain being fed into the motor model. The problem with using a human as a controller is the reaction time we have in adjusting the gain. In a simulation of only five seconds the accuracy of the manual control will depend on the individual. We also introduce the need to use a filter on our output since the noise from position encoder is amplified when we take the discrete derivative.

II. Procedure

The first step of the procedure was to download the Simulink block from blackboard. The Simulink block is shown below.



i

Figure 1: Simulink Model for Manual Control

Once the block was set up we had to find the top speed for our motor by adjusting the slider gain and setting the normalization gain to 1. Once we identified the top speed of our motor, we normalized it by setting the normalization gain, b, to $\frac{1}{top\ speed}$. Then we configured the low pass filter block as follows.

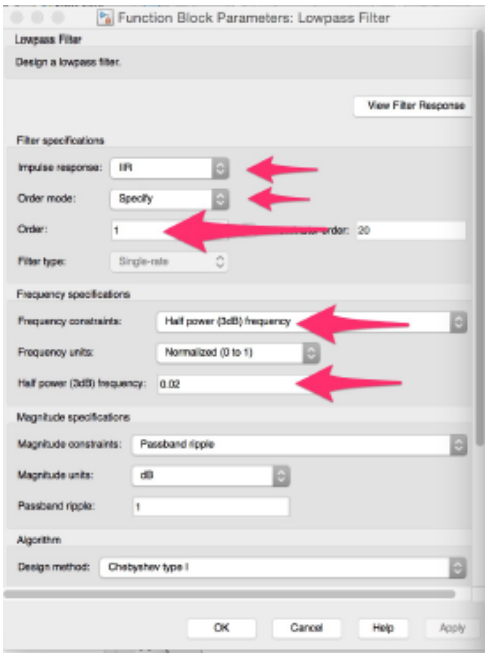


Figure 2: Low Pass Filter Parameters

Then we started the simulation and attempted to get the top speed to settle at .5 by adjusting the slider gain.

Then we attempted to do the same thing by using a PI controller designed in parallel. The block diagram is shown below.

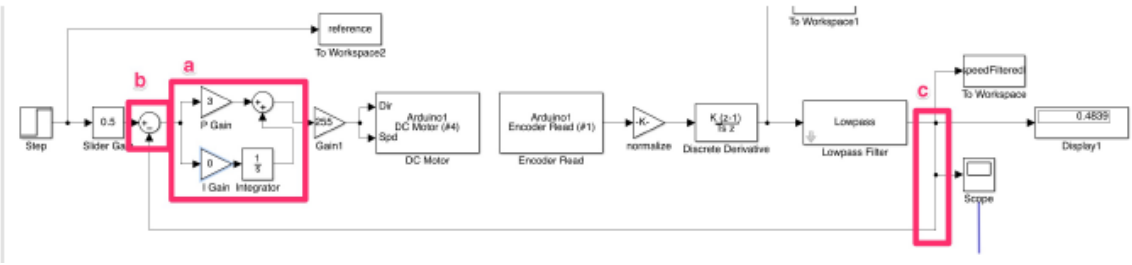


Figure 3: Simulink Model for PI Control

Once the block was configured, we started testing by setting the P gain to 1 and the I gain to 0. We adjusted the P and I gains in attempting to get the best possible output. The results are shown in the Results section of the report.

III. Results

Before we applied the low pass filter, the output of the motor looked like what is seen below in Figure 4.

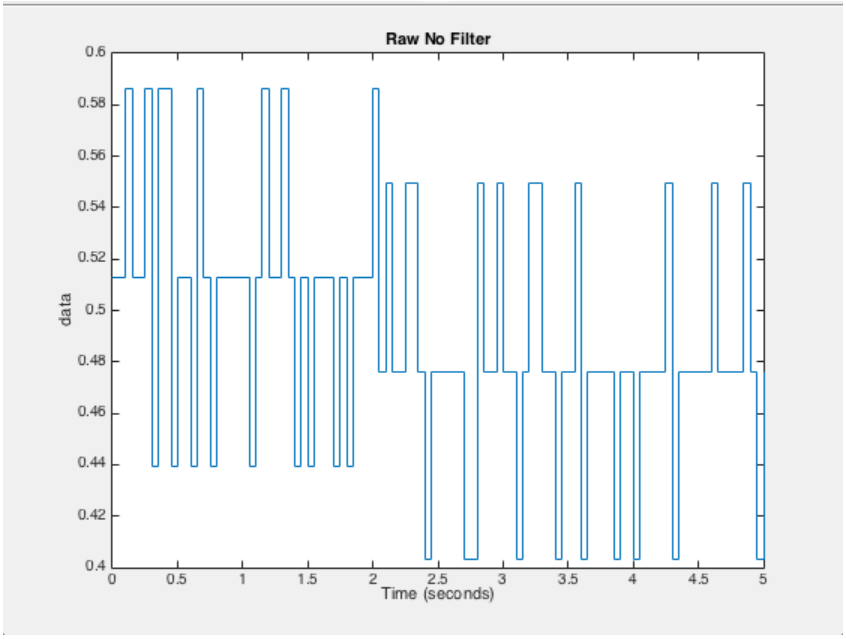


Figure 4: Manual control Output not Filtered

After applying the low pass filter, we see the large noise components are cut out, leaving the following output.

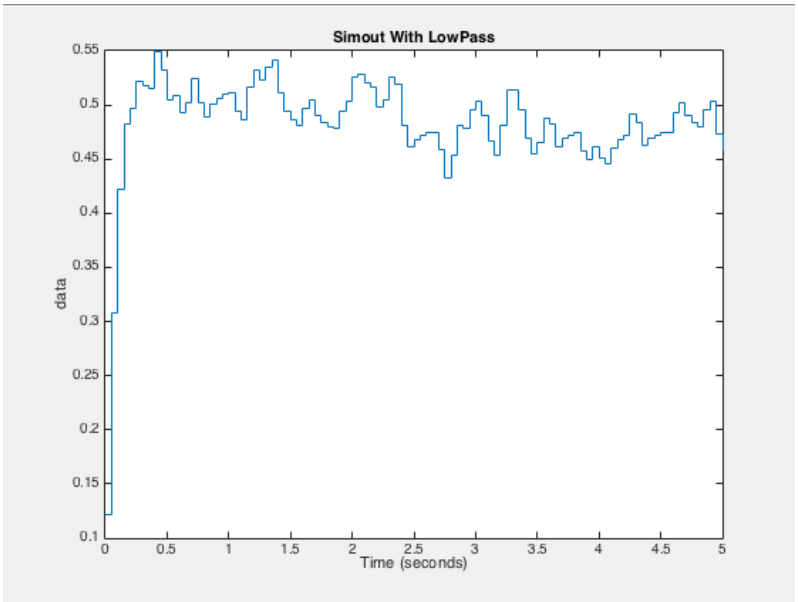


Figure 5: Manual control output Filtered

We can see that with the slider gain set to about .5, our output also settles around .5.

Now we had to use the PI controller to achieve similar results. The first test was when P=1 and I=0. This yielded the following results.

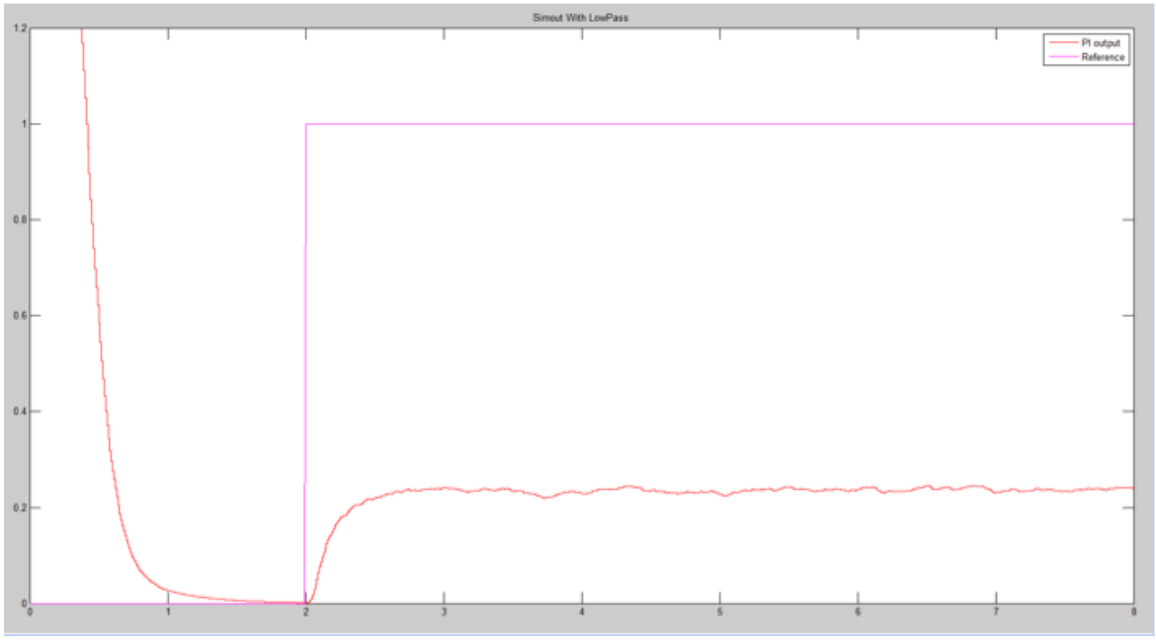


Figure 6: P = 1 I = 0

With this configuration, we see that the proportional gain, which mainly controls the transient response, produces a fast rise time, but we never reach a steady state value of .5. For this reason, we increased P to 10 and I to .5. This yielded the following output.

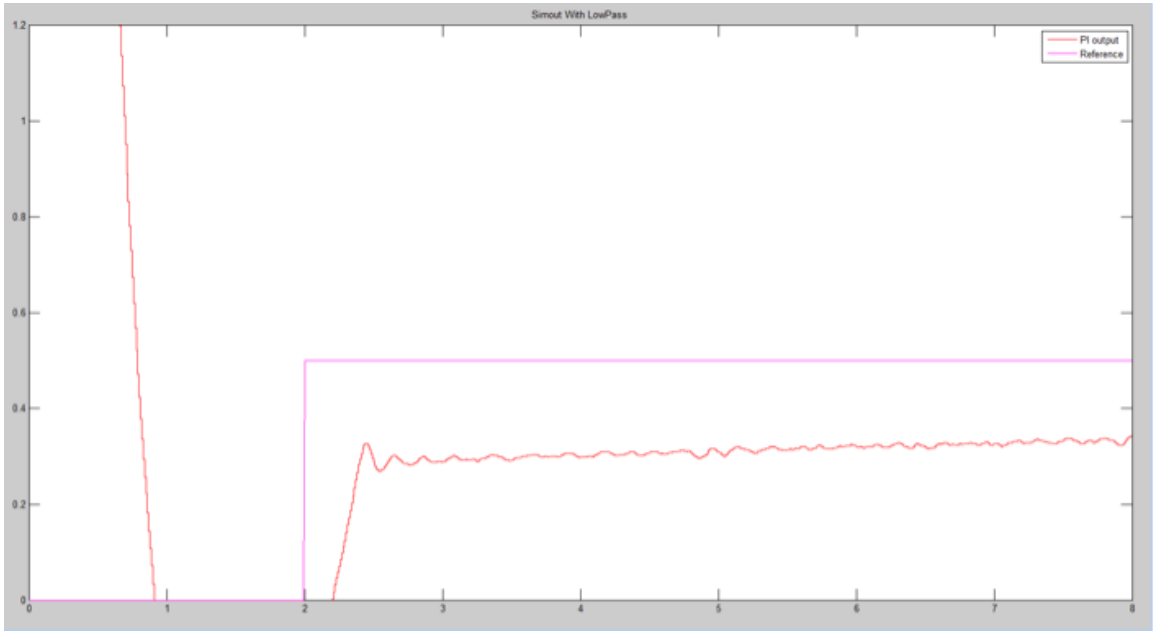


Figure 7: P = 10 I = .5

Here we can see that our system is almost the same as the previous output, except the response oscillates slightly. Clearly, this is not the combination of P and I that will achieve our desired steady state output. The rise time appears to be better this time however. For this reason, we chose to keep the P gain at 10 but lower the I gain to .01. This finally yielded the result we were after, which is shown below.

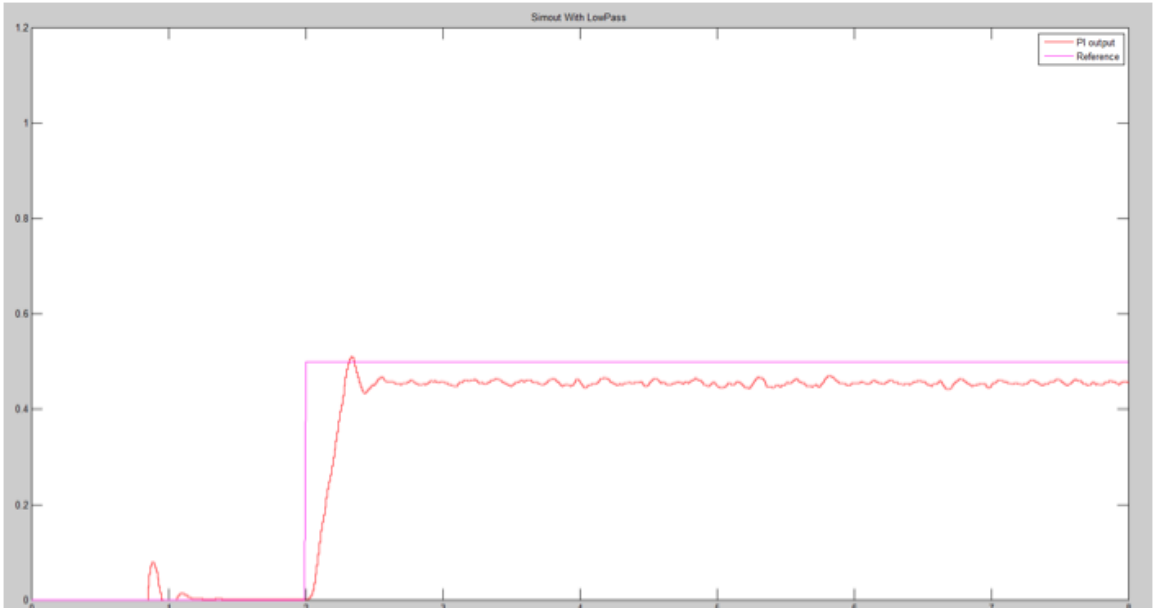


Figure 8: P = 10 I = .01

In this figure we see a very small steady state, a high rise time, and minimum oscillation in the output, which tells us that the gains set for P and I were good for this application.

IV. Conclusion/Discussion

The challenge is over and machine has been crowned the victor! On a serious note, we did see the PI controller perform better overall than our hand controlled gain. It took us serval attempts in order to observe a gain that would produce an output we desired. The first problem we ran into was a noisy output, which comes from the discrete derivative of our encoder. We resolved the issue by using a low pass filter on the output. The low pass filter will cut off any higher frequencies in our signal which is where the noise resides. After realizing quickly how impractical a manual controller is we switched to the PI controller and observed more realistic results.

The PI controller we developed was contrasted using individual proportional and integral gains blocks in Simulink. The parallel design allows each components (P and I) to see the error of the feedback network. After we find the proportional and integral gain of our error we sum them back up to gain the input to our motor model. If we included a derivative gain it would amplify the noise in our output. Using only the integrator and proportional blocks produce a good controller. The proportional will feed the error directly into the motor block. The integrator block will take an accumulation of the total error over a period of time and feed that gain into our motor block. Having a proportional gain that is too low produce a response that settled at a speed that was too slow. Too high and the speed of our motor was too fast. After finding a proportional gain that produce relatively good results we added our integral gain. Using an integral gain that was too high increased our settling time.

In the end the PI controller was more practical implementation of a controller. Our manual control only received good results after several trial and error cases. In a real time system however our trial and error method could not be implemented. The PI controller responds in real time to the system if configured correctly. There is a reason we do not see humans advertising themselves as professional system controllers.